数据可视化实验报告

计算机科学与技术学院

班　　级：大数据 2101 班

学　　号：U202115652

姓　　名：李嘉鹏

指导教师：何云峰

完成日期：2023 年 12 月

实验报告及设计评分细则

评 分 项 目	满分	得分	备注	
文档格式（段落、行间距、缩进、图表、编号等）	15			实 验 报 告 总分
实验方案设计	10			
实验过程	50			
遇到的问题及处理	10			
设计方案存在的不足	5			
心得（含思政）	5			
意见和建议	5			
可视化作品	100			
教师签名			日　　期	

备注：实验过程将从可视化作品的完成度、是否讲述好数据的故事、作品的复杂度等方面进行综合评分。

实验课程总分=可视化*0.6+实验报告*0.4

目　录

1 实验概述

1.1 实验名称

设计一个数据可视化作品，讲述数据的故事。

1.2 实验目的

（1）了解数据可视化方案的设计和实现方法；

（2）利用 python 的数据可视化工具库（matplotlib 库、pyecharts 库）进行可视化方案的实现。

1.3 实验环境

软件：python。

可用库：数据处理库 NumPy、pandas；可视化库 matplotlib、pyecharts 库等。

可视化视图：pyecharts 中提供了 Bar(柱状图/条形图)；Bar3D(3D 柱状图)；Boxplot(箱形图)；EffectScatter(涟漪散点图)；Funnel (漏斗图)；Gauge(仪表盘)；Geo(地理坐标系)；Graph(关系图)；HeatMap(热力图)；Kline(K 线图)；Line(折线/面积图)；Line3D(3D 折线图)；Liquid(水球图)；Map(地图)；Parallel (平行坐标系；Pie(饼图)；Polar(极坐标系)；Radar(雷达图)；Sankey(桑基图)；Scatter(散点图等，可以根据需要进行选择。

1.4 实验内容

（1）在备选的 4 个数据集中任意选择其中一个数据集，对数据集中的数据进行数据分析和预处理，设计合理的可视化方案。

（2）利用 python 的可视化工具库实现可视化方案。

（3）完成实验报告，讲述数据的故事。

1.5 实验要求

（1）对数据集中的数据进行分析和整理，提取出其中的有效信息，完成数据的预处理；

（2）设计合理的可视化视图，选择合适的视觉通道，表达数据的某个特性；

（3）完成不少于 6 个可视化视图，较为完整地描述数据集。

2 总体设计

2.1 总体设计思路

本实验中我选择的数据集为世界杯数据集。

世界杯数据集主要由三个数据表构成，分别是 WorldCupsSummary.csv、WorldCupMatches.csv 和 WorldCupPlayers.csv。下面将对数据集进行简要的说明。

（1）世界杯成绩信息表：WorldCupsSummary.csv

包含所有 21 届世界杯赛事（1930-2018）的比赛主办国、前四名队伍、总参赛队伍、总进球数、现场观众人数等信息。包括如下字段：

字段名称	含义	字段类型
Year	举办年份	数值型
HostCountry	举办国家	字符型
Winner	冠军队伍	字符型
Second	亚军队伍	字符型
Third	季军队伍	字符型
Fourth	第四名队伍	字符型
GoalsScored	总进球数	数值型
QualifiedTeams	总参赛队伍数	数值型
MatchesPlayed	总比赛场数	数值型
Attendance	现场观众总人数	数值型
HostContinent	举办国所在洲	字符型
WinnerContinent	冠军国家队所在洲	字符型

（2）世界杯比赛比分汇总表：WorldCupMatches.csv

包含所有 21 届世界杯赛事（1930-2014）单场比赛的信息，包括比赛时间、比赛主客队、比赛进球数、比赛裁判等信息。包括如下字段：

字段名称	含义	字段类型
Year	比赛（所属世界杯）举办年份	数值型
Datetime	比赛具体日期	日期型
Stage	比赛所属阶段，包括小组赛（GroupX）、16 进 8（Quarter-Final）、半决赛（Semi-Final）、决赛（Final）等	字符型

Stadium	比赛体育场	字符型
City	比赛举办城市	字符型
Home Team Name	主队名	字符型
Away Team Name	客队名	字符型
Home Team Goals	主队进球数	数值型
Away Team Goals	客队进球数	数值型
Attendance	现场观众数	数值型
Half-time Home Goals	上半场主队进球数	数值型
Half-time Away Goals	上半场客队进球数	数值型
Referee	主裁	字符型
Assistant 1	助理裁判 1	字符型
Assistant 2	助理裁判 2	字符型
RoundID	比赛所处阶段 ID，和 Stage 字段对应	数值型
MatchID	比赛 ID	数值型
Home Team Initials	主队名字缩写	字符型
Away Team Initials	客队名字缩写	字符型

（3）世界杯球员信息表：WorldCupPlayers.csv

包含世界杯历史上球员的相关信息，包括如下字段：

字段名称	含义	字段类型
RoundID	比赛所处阶段 ID，同比赛信息表的 RoundID 字段	数值型
MatchID	比赛 ID	数值型
Team Initials	队伍名	字符型
Coach Name	教练名	字符型
Line-up	首发/替补	字符型
Shirt Number	球衣号码	数值型
Player Name	队员名	字符型
Position	比赛角色（包括 C=Captain，GK=Goalkeeper）	字符型
Event	比赛事件（包括进球、红/黄牌等）	字符型

WorldCupsSummary.csv 数据集共包含 22 条数据，WorldCupMatches.csv 数据集共包含 853 条数据，WorldCupPlayers.csv 数据集共包含 37785 条数据。

2.2 总体设计框架

我对世界杯数据集的总体设计框架包括数据分析、数据预处理、可视化视图选择与实现、用户交互设计四个部分，最终的可视化方案共包括 9 个不同的视图，全方面展现数据的特征。

• 数据分析

首先分析世界杯数据集的数据特征，观察是否出现明显噪声数据（便于后续进行数据预处理），并根据数据特点和内容草拟出可视化设计方案。

• 数据预处理与数据变换

检测数据中存在的噪声和错误，选择合适的数据预处理方法提高数据质量，如数据清洗、异常值与缺失值的填充等。同时，还需要对特定类型的数据进行基本的数据变换，例如归一化等操作。

• 可视化编码、视图选择与实现

在第一步中，已经对数据的特点进行了初步分析，此处需要进一步选择合适的可视化编码与视图并展示数据，需要确保将数据以一种直观易懂的方式呈现出来，提高数据的精确性和可辨性。

对于可视化编码，主要需要考虑标记（图形元素）和视觉通道（位置、大小、形状、方向、颜色等）两个维度。

对于视图，也有很多种可行的选择，包括趋势型图表（折线图、面积图）、对比型图表（柱状图、雷达图）、构成型图表（饼图、环形图）、分布型图表（散点图、气泡图、直方图、热力图）等。

• 用户交互设计

为了让图“动起来”，可以引入一些用户交互手段以增强图形的丰富度。常见的用户交互方式包括滚动、缩放、选择、过滤等，这可以提高可视化方案中根据用户需求进行数据查询的自由度。

综上所述，本实验的总体设计框架如图 2.1 所示。

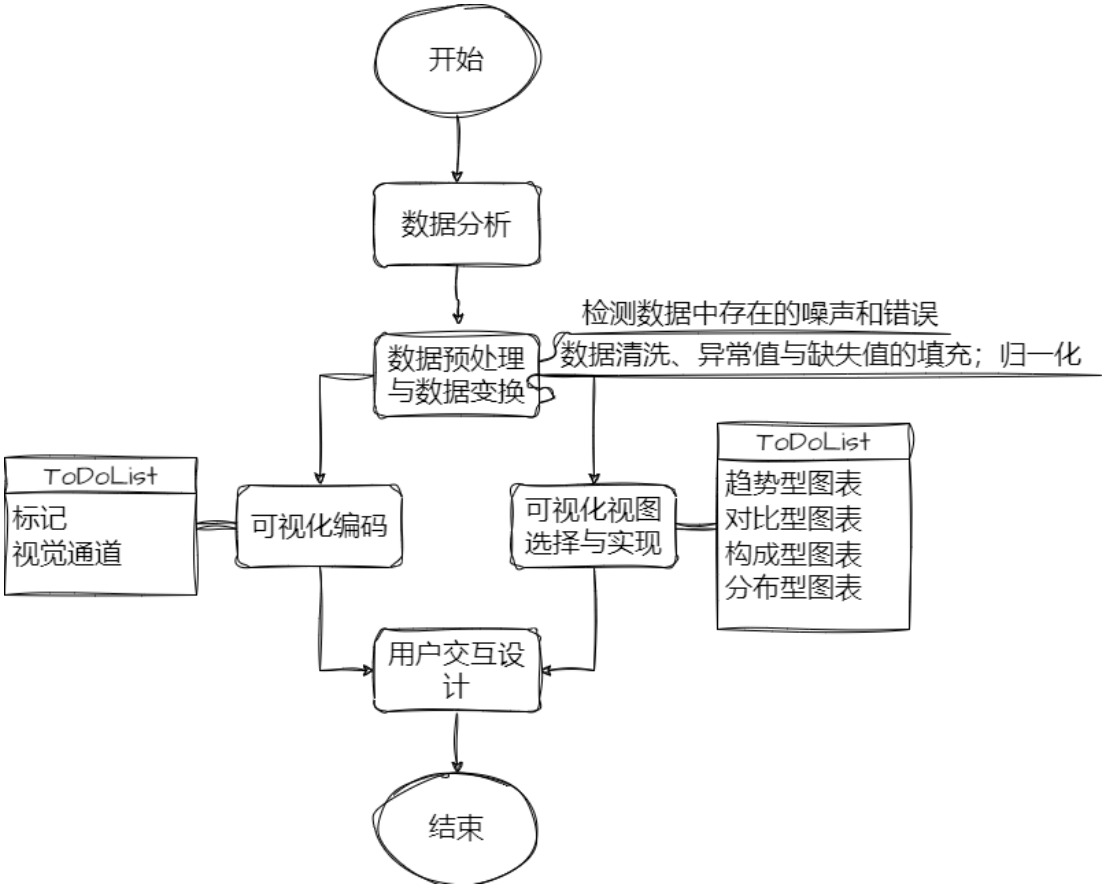

图 2.1：总体设计框架示意图

3 实验过程

3.1 数据预处理

（1）数据预处理设计

首先，粗略观察上述三个数据集。

• 对于 WorldCupsSummary.csv 数据集，其数量条数很少，因此可以直接看出其不存在数据缺少和异常的问题。数据特征如图 3.1 所示。

```
Year,HostCountry,Winner,Second,Third,Fourth,GoalsScored,QualifiedTeams,MatchesPlayed,Attendance,HostContinent,W
1930,Uruguay,Uruguay,Argentina,USA,Yugoslavia,70,13,18,590549,America,America
1934,Italy,Italy,Czechoslovakia,Germany,Austria,70,16,17,363000,Europe,Europe
1938,France,Italy,Hungary,Brazil,Sweden,84,15,18,375700,Europe,Europe
1950,Brazil,Uruguay,Brazil,Sweden,Spain,88,13,22,1045246,America,America
1954,Switzerland,Germany FR,Hungary,Austria,Uruguay,140,16,26,768607,Europe,Europe
1958,Sweden,Brazil,Sweden,France,Germany FR,126,16,35,819810,Europe,America
1962,Chile,Brazil,Czechoslovakia,Chile,Yugoslavia,89,16,32,893172,America,America
1966,England,England,Germany FR,Portugal,Soviet Union,89,16,32,1563135,Europe,Europe
1970,Mexico,Brazil,Italy,Germany FR,Uruguay,95,16,32,1603975,America,America
1974,Germany,Germany FR,Netherlands,Poland,Brazil,97,16,38,1865753,Europe,Europe
1978,Argentina,Argentina,Netherlands,Brazil,Italy,102,16,38,1545791,America,America
1982,Spain,Italy,Germany FR,Poland,France,146,24,52,2109723,Europe,Europe
1986,Mexico,Argentina,Germany FR,France,Belgium,132,24,52,2394031,America,America
1990,Italy,Germany FR,Argentina,Italy,England,115,24,52,2516215,Europe,Europe
1994,USA,Brazil,Italy,Sweden,Bulgaria,141,24,52,3587538,America,America
1998,France,France,Brazil,Croatia,Netherlands,171,32,64,2785100,Europe,Europe
2002,Korea/Japan,Brazil,Germany,Turkey,Korea Republic,161,32,64,2705197,Asia,America
2006,Germany,Italy,France,Germany,Portugal,147,32,64,3359439,Europe,Europe
2010,South Africa,Spain,Netherlands,Germany,Uruguay,145,32,64,3178856,Africa,Europe
2014,Brazil,Germany,Argentina,Netherlands,Brazil,171,32,64,3386810,America,Europe
2018,Russia,France,Croatia,Belgium,England,169,32,64,3031768,Europe,Europe
```

图 3.1：WorldCupSummary 数据特征

• 对于 WorldCupMatches.csv 数据集，抽取其中一条数据如下所示：

```
1930.0,26 Jul 1930 - 14:45 ,Semi-finals,Estadio
Centenario,Montevideo ,Argentina,6.0,1.0,USA,72886.0,1.0,0.0,LANGENUS
Jean (BEL),VALLEJO Gaspar (MEX),WARNKEN Alberto
(CHI),202.0,1088.0,ARG,USA
```

可以发现对于 Year 字段，理论上应该为整型变量，但此处为浮点型变量，需要统一进行修改。对于其它字段，肉眼无法直接观察到数据缺失，因此可以写一个检验脚本程序，判定是否存在数据缺失的现象发生。

• 对于 WorldCupPlayers.csv 数据集，未发现明显问题，同样使用脚本检验是否存在缺失值（只需要检验前 8 列，最后一列 event 可以为空）。

（2）数据预处理的实现

•对于 WorldCupMatches.csv 数据集中的 Year 字段，通过以下代码将其转换为 int 型变量：

```
'Year' = df_clean['Year'].astype(int)
```

检验数据是否存在缺失的脚本程序如下，需要引入 pandas 库，并遍历每一行检查是否有空值：

```
import pandas as pd
df = pd.read_csv('WorldCupMatches.csv')
missing_fields = df[df.isnull().any(axis=1)].index
for row_number in missing_fields:
   print("缺失字段的行号：", row_number)
```

运行上面的代码后，可以发现 WorldCupMatches.csv 和 WorldCupPlayeres.csv 数据集不存在数据缺失的问题，因此不需要进行额外处理。

3.2 分组柱状图视图：世界杯进球数和参赛队伍数柱状图

（1）设计思路及设计过程

柱状图（或条形图）有利于直观地展示数据之间的对比。为了在一张图内同时描述世界杯历年的进球数和参赛队伍数，我采用了分组柱状图的方式，其中横轴代表年份（从 1930-2018 升序排序），每个年份对应进球数和参赛队伍数两个属性的数据；纵轴代表数量。通过对比每个柱子的高度，可以很明显地看出数量的多少。

可视化编码：本视图的图形元素为二维“面”，采用的视觉通道包括互异的位置、蓝色和绿色的颜色、垂直的方向、矩形的形状。

代码实现：首先调用 Bar()类初始化了一个对象 bar，使用 add_xaxis()和 add_yaxis()方法分别将世界杯数据中的年份、进球数和参赛队伍数数据添加到对应的轴上，同时设置了相应的标签。随后调用 set_global_opts()方法设置了柱状图的全局选项，包括标题和数据缩放选项（slider）。

本视图如图 3.2 所示。

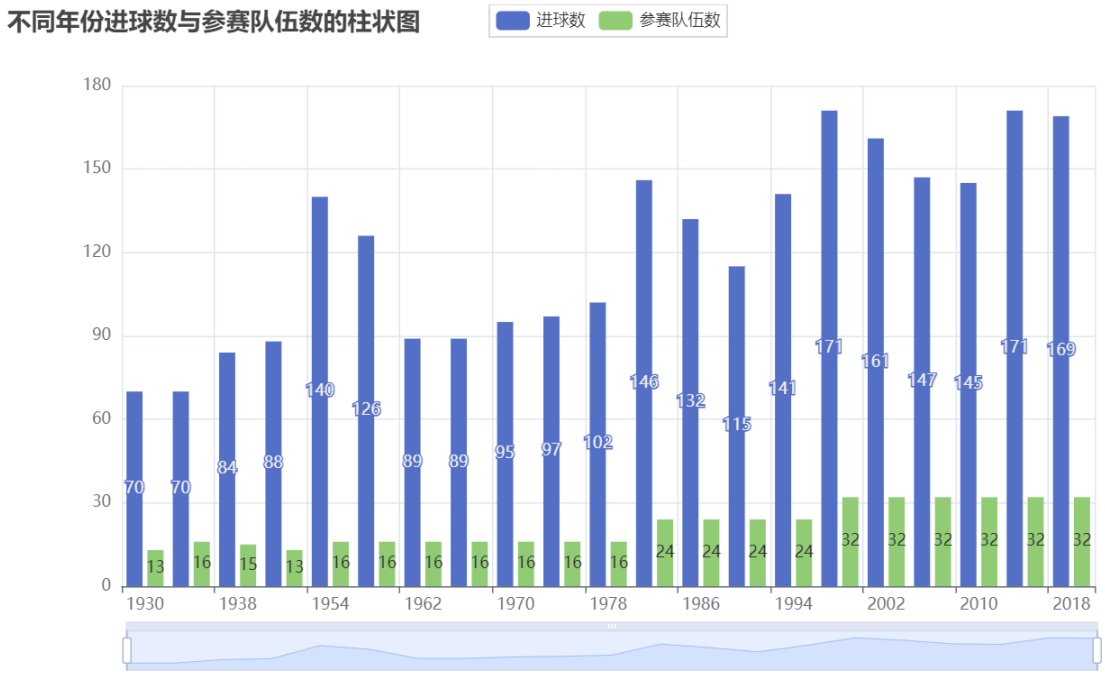

图 3.2：分组柱状图视图——世界杯进球数和参赛队伍数柱状图

（2）交互设计

本视图主要的交互方法包括交互式工具箱和数据范围缩放条。

其中，交互式工具箱位于视图的上方，用户可以点击蓝色和绿色的色块（分别代表进球数和参赛队伍数）自行选择展示何种数据，为用户提供了一定自由度，便于根据实际需求针对性地进行数据查询。

数据范围缩放条位于视图的下方，用户可以自行调整左边界和右边界选定横轴的范围，也可以拖动缩放条切换横轴范围。

（3）视图分析

该分组柱状图（条形图）展示了世界杯历年的进球数和参赛队伍数，并按照年份从远到近对数据进行了整理。从图中可以看出，世界杯历史上每年的进球数不断波动，但总体趋势还是缓慢提高，2018 年的进球数是 1930 年进球数的两倍多；而每年的参赛队伍数的规律性很强，在 1930~1978 年的区间内参赛队伍数一般都不超过 16，在 1982~1994 年的区间内参赛队伍数均为 24，在 1998 年以后的区间内参赛队伍数均为 32，这说明世界杯的赛制可能在 1982 和 1998 年发生了两次重要变化。

3.3 折线图视图：观众人数和比赛场次随时间变化趋势折线图

（1）设计思路及设计过程

折线图的特点是能十分直观地反映单个或多个指标随时间的变化趋势。对于世界杯数据集，通过采用双轴折线图的方式，可以在一张图上清晰地展示出观众人数和比赛场次随世界杯年份的变化，其中横轴代表年份（从 1930-2018 升序排序），每个年份对应观众人数和比赛场次两个属性的数据；纵轴分为左轴和右轴，左轴为观众人数，右轴为比赛场次。通过观察观众人数和比赛场次这两条折线（蓝色线和绿色线）的走向，可以很明显地看出二者的发展趋势。

可视化编码：折线图包括了两个折线图形元素，采用了不同的颜色和线型来区分。本视图的图形元素包括零维“点”和一维“线”，采用的视觉通道包括互异的位置、蓝色和绿色的颜色以及相邻同类样本点之间的连接关系。

代码实现：首先通过 Line()类初始化了一个对象 line，并使用 add_xaxis()和 add_yaxis()方法添加了观众人数和比赛场次的数据，由于两种属性的数据不在同一数量级，因此需要分别设置左轴和右轴（调用 extend_axis()方法）。随后调用 set_global_opts()方法设置了全局选项，包括标题、坐标轴命名和数据缩放选项（slider）以及交互提示信息（tooltip，使鼠标移动到图中时产生十字对齐效果）。

本视图如图 3.3 所示。

图 3.3：折线图视图——观众人数和比赛场次随时间变化趋势折线图

（2）交互设计

本视图主要的交互方法包括交互式工具箱、数据范围缩放条和交互提示信息。

其中，交互式工具箱位于视图的上方，用户可以点击蓝色和绿色的曲线（分别代表观众人数和比赛场次）自行选择展示何种数据，便于根据实际需求针对性地进行数据查询。

数据范围缩放条位于视图的下方，用户可以自行调整左边界和右边界选定横轴的范围，也可以拖动缩放条切换横轴范围。

交互提示信息可在鼠标悬停时展示具体的数据数值，由于此处是十字交叉对齐的效果，方便用户准确锁定目标数据点并获取信息。

用户同时使用交互式工具箱、数据范围缩放条和交互提示信息的效果如图 3.4 所示。

图 3.4：折线图视图——观众人数和比赛场次随时间变化趋势折线图用户交互效果

（3）视图分析

该折线图展示了世界杯观众人数和比赛场次随年份的变化趋势，并按照年份从远到近对数据进行了整理，能直观地比较它们之间的关联情况、具体查询某一年的观众人数和比赛场次数据，也可以只保留一类数据进行查询。可以发现，观众人数与比赛场次都有不断升高的趋势，且二者的增长趋势存在一定相关性。

3.4 面积图视图：观众人数和比赛场次随时间变化趋势面积图

（1）设计思路及设计过程

面积图与折线图类似，主要通过折线下方面积的变化判定某属性的变化趋势。在上一节折线图视图的基础上，采用了透明度为 0.5 的面积样式，使多数据线之间的叠加更清晰。

可视化编码：本视图的图形元素为零维“点”、一维“线”和二维“面”，采用的视觉通道包括互异的位置、蓝色和绿色的颜色、不同区域的亮度以及相邻同类样本点之间的连接关系。

代码实现：调用 Line()类初始化一个对象 line，使用 add_xaxis()和 add_yaxis()方法分别将观众人数和比赛场次数据添加到对应的轴上，同时调用 extend_axis()方法添加了右侧的纵轴。随后调用 set_global_opts()方法设置了全局选项，包括标题、坐标轴命名和数据缩放选项（slider）以及交互提示信息（tooltip，使鼠标移动到图中时产生十字对齐效果）。最后将两条折线对应的面积重叠透明度设为 0.5。

本视图如图 3.5 所示。

图 3.5：面积图视图——观众人数和比赛场次随时间变化趋势面积图

（2）交互设计

本视图主要的交互方法包括交互式工具箱、数据范围缩放条和交互提示信息。具体设计与 3.3 节一致，此处不再赘述。

（3）视图分析

该面积图展示了世界杯观众人数和比赛场次随年份的变化趋势，能直观地比较它们之间的关联情况、具体查询某一年的观众人数和比赛场次数据，也可以只保留一类数据进行查询。可以发现，观众人数与比赛场次都有不断升高的趋势，且二者的增长趋势存在一定相关性。

3.5 环形图视图：世界杯结果分布环形图

（1）设计思路及设计过程

环形图属于饼图的一种，最适合展示各类数据在总体中的分布情况。本视图使用了圆环图展示世界杯历年的冠军、亚军、季军和第四名分布情况。圆环图中各种颜色所占的比例，清晰展现了不同队伍在世界杯历史上的表现。

可视化编码：本视图的图形元素为二维“面”，即环形图中的扇形区域；采用的视觉通道包括互异的角度、丰富的色调（红橙黄绿青蓝紫）和多样化的亮度（如深绿和浅绿）。

代码实现：首先创建四个圆环图对象（pie1, pie2, pie3, pie4）。通过 add()方法将世界杯历年比赛结果数据添加到圆环图中，并区分四种不同类型的结果。随后使用 set_series_opts()方法对标签进行格式化，突出显示队伍名称和数量，并通过 set_global_opts()方法设置图表的标题和图例位置，提升了图表的可读性和易懂性。为了将这四张图放置在一个 html 文件中，我创建了一个页面对象 page，并将四个 pie 对象添加到页面中，最后只需要渲染整个页面即可。

本视图针对冠军、亚军、季军、第四名的结果分别如图 3.6~图 3.9 所示。

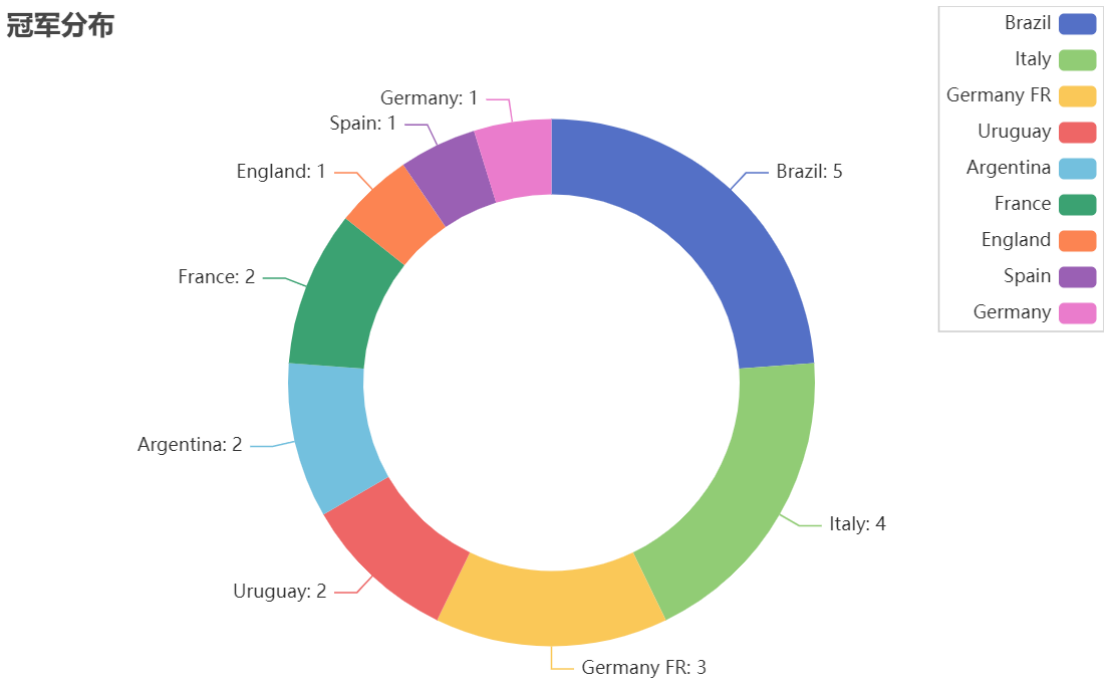

图 3.6：环形图视图——世界杯结果分布环形图（冠军分布）

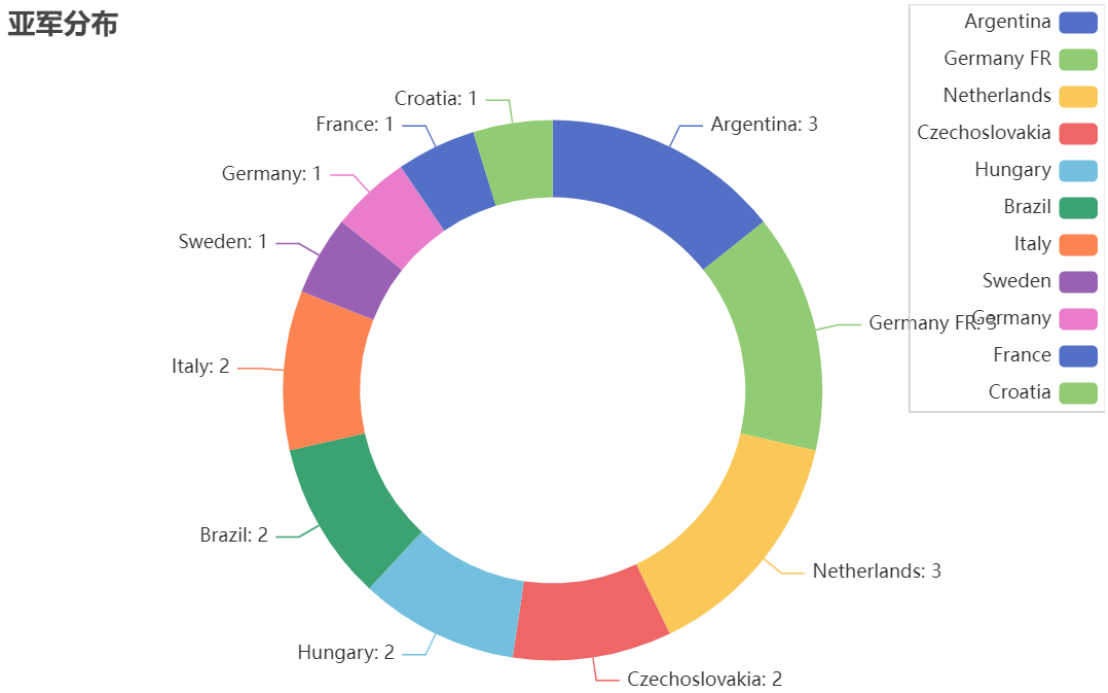

图 3.7：环形图视图——世界杯结果分布环形图（亚军分布）

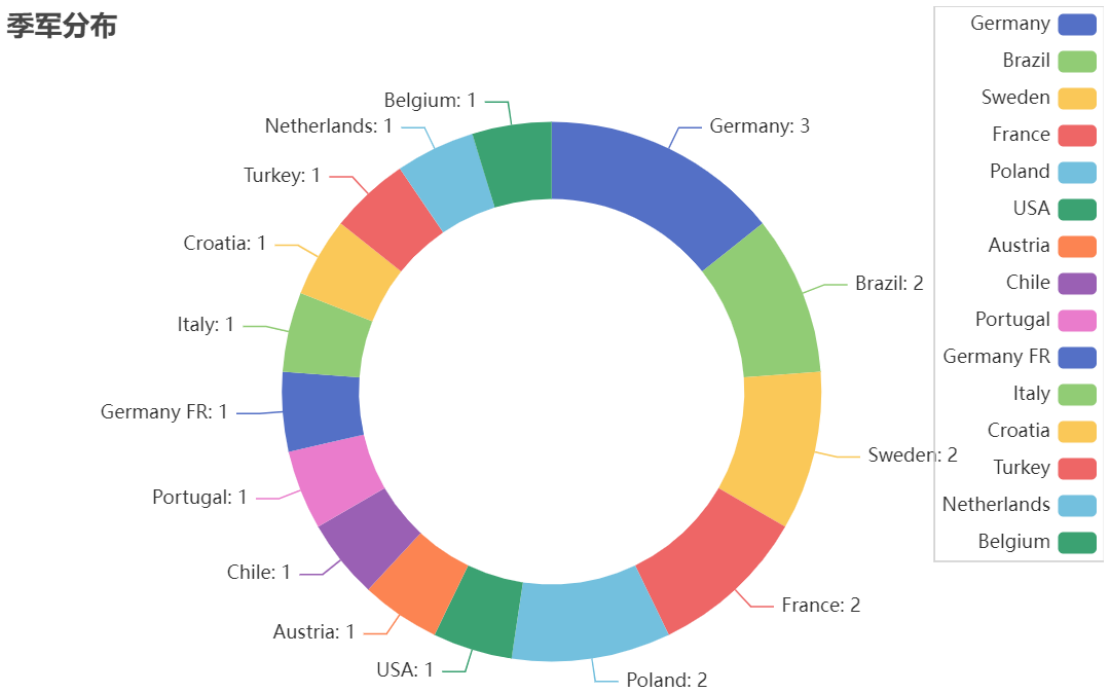

图 3.8：环形图视图——世界杯结果分布环形图（季军分布）

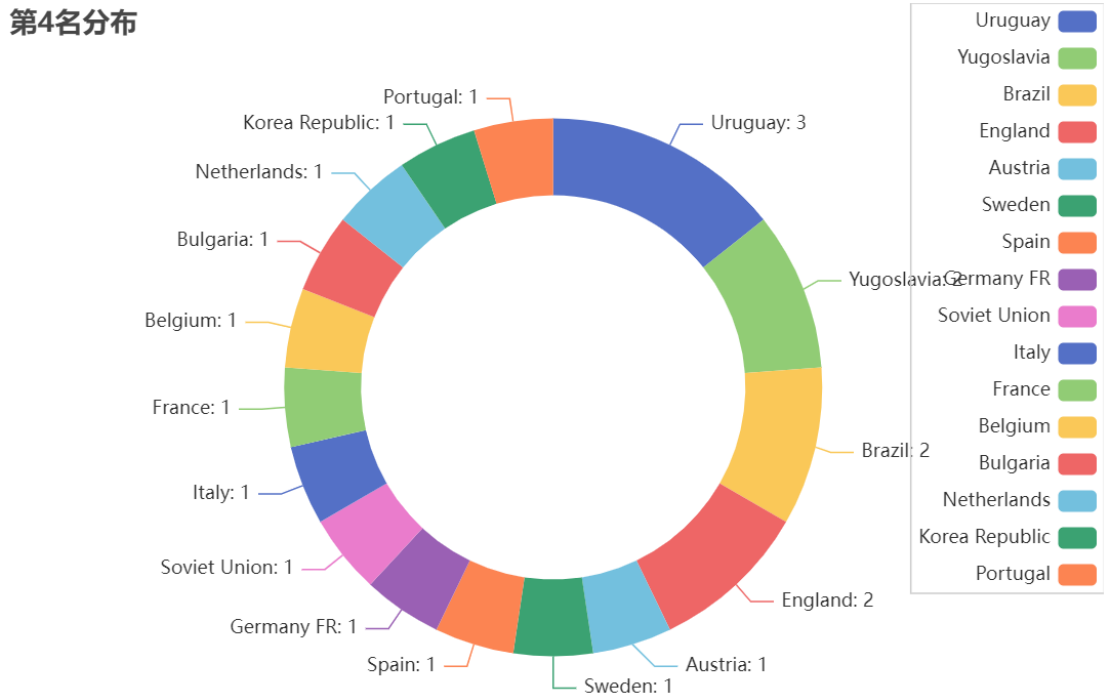

图 3.9：环形图视图——世界杯结果分布环形图（第四名分布）

（2）交互设计

本视图的交互方法主要是交互式工具箱。交互式工具箱位于视图的右侧，通过点击图例中不同的队伍名称，用户可以自行选择展示何种数据，从而方便用户根据不同的需求进行针对性的结果查询。

（3）视图分析

该环形图清晰呈现了世界杯冠军、亚军、季军和第四名的分布情况。从图中可以观察到不同队伍在世界杯历史上的表现，为用户提供了直观的认知，便于比较和分析不同队伍的水平，进一步了解世界杯赛事的历史数据。

以冠军分布为例（如图 3.6），巴西获得世界杯冠军的次数最多，为 5 次；紧随其后的是意大利 4 次、德国 3 次等，据此可以得出巴西足球实力强大的结论。同样地，对于亚军、季军和第四名的分布，用户也可以单独查阅，从而找出在世界杯历史上获得过突出成绩的国家队。

可见，通过环形图，用户可以直观了解不同队伍在世界杯历届比赛中的表现，为用户提供了直观的数据展示。

3.6 多层级饼图（南丁格尔玫瑰图）视图：世界杯结果分布饼图

（1）设计思路及设计过程

多层级饼图是饼图的一种高级形式，适合展示具有层级关系的数据在总体中的分布情况。为了把冠军、亚军、季军和第四名的分布情况放在一张图中，我使用了四个多层级饼图，分别展示了世界杯历年前四名的分布情况。通过多层级饼图，用户可以直观地比较不同队伍在世界杯历史上的表现，呈现了数据的层级关系和分布情况。

在饼图的基础上，我还融入了南丁格尔玫瑰图的思路，给定条件下出现的次数越多，相应的扇形半径越大，反之半径越小，这样可以增强视图的视觉冲击力。

可视化编码：本视图的图形元素为二维“面”，即多层级饼图中的各层级区域；采用的视觉通道包括不同半径的层级嵌套、多样的色调和不同的亮度，使得数据分布更加清晰。

代码实现：创建四个多层级饼图对象（pie1, pie2, pie3, pie4），将世界杯历年的比赛结果数据添加到各个多层级饼图中，并区分四种不同类型的结果。随后将玫瑰图的种类 rosetype 设为半径 radius，然后利用 set_series_opts()方法对标签进行格式化，突出显示队伍名称和数量，并通过 set_global_opts()方法设置图表的标题和图例位置。最后，使用 Grid()将四张饼图放置在一个子图中。

本视图如图 3.10 所示。

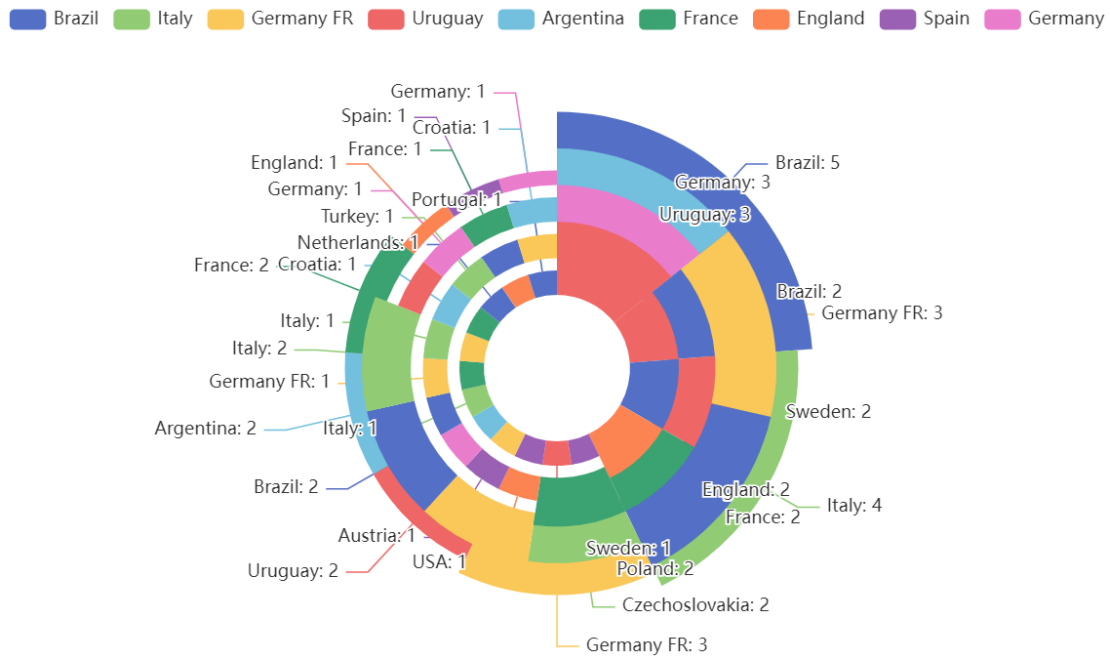

图 3.10：多层级饼图（南丁格尔玫瑰图）视图——世界杯结果分布饼图

（2）交互设计

本视图主要的交互方法包括交互式工具箱和交互提示信息。

其中，交互式工具箱位于视图的上方，用户可以点击各个色块自行选择展示哪些国家的数据，便于根据实际需求针对性地进行数据查询。

交互提示信息可在鼠标悬停时展示具体的数据数值，并选择特定层级的数据进行查看，提供了交互式演示的能力。

（3）视图分析

与上面的环形图相比，多层级饼图（南丁格尔玫瑰图）以不同半径的层级完成了四个饼图的嵌套，使得数据分布更加清晰，用户可以更直观地了解数据的层级关系和分布情况。从图中可以清晰地观察到冠军、亚军、季军和第四名在世界杯历年赛事中的分布情况，为用户提供了直观的历史数据展示，便于比较和分析不同队伍的表现。

3.7 地图视图：世界杯举办国家分布地图

（1）设计思路及设计过程

地图适用于展示地域之间在某属性上的差异。为了表现历史上世界杯举办地域的分布，我使用地图视图展示了世界杯在不同国家举办的次数分布情况。通过比对不同地区的颜色深浅，可以快速得知其举办世界杯的次数。图中，举办过 0

次、1 次、2 次世界杯的国家的颜色分别为白色、绿色、橙色。

可视化编码：本视图的图形元素为二维“面”，即将数据映射到世界地图上；采用的视觉通道包括不同的位置、不同的区域大小以及多样的色调（白色、绿色、橙色），以提高地图的可读性和易懂性。

代码实现：首先创建一个 Map 类对象 map，加载数据后使用 value_counts()方法统计每个国家作为世界杯主办国的次数，并将结果保存到 host_count 变量中。接下来通过 Map()函数的 init_opts 参数设置了地图的初始选项，使用 add()方法将世界杯举办次数的数据添加到地图中。这里使用了 zip()函数将主办国和次数两个列表合并为一个新的列表作为 add()方法的输入数据。在 set_global_opts()方法中设置了地图的全局选项，包括标题和视觉映射选项。

本视图如图 3.11 所示。

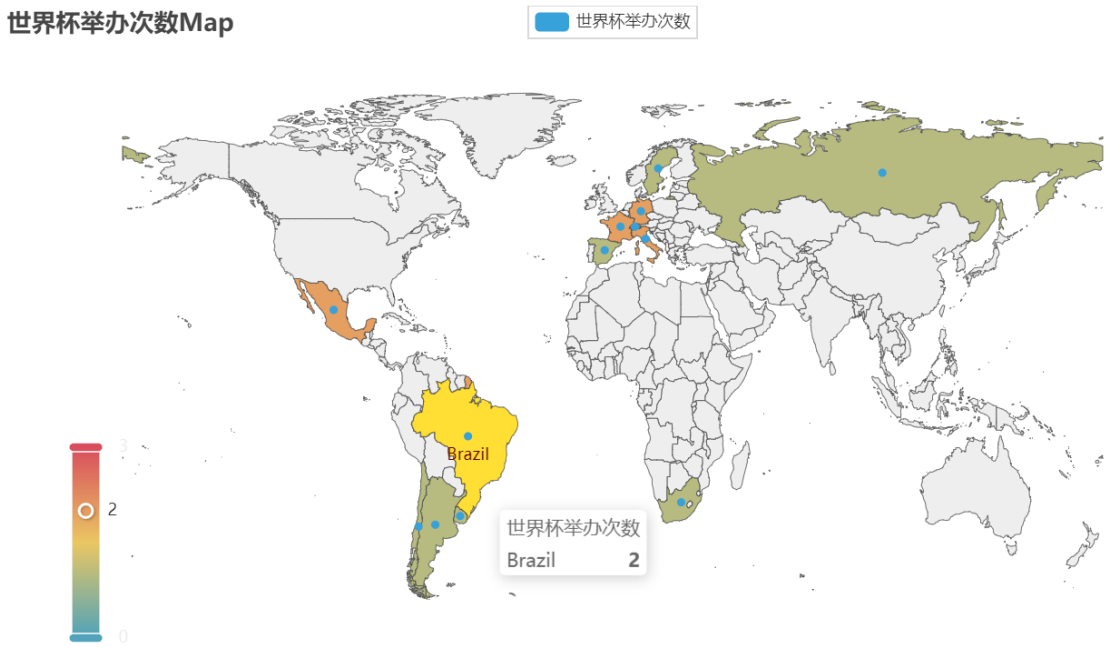

图 3.11：地图视图——世界杯举办国家分布地图

（2）交互设计

本视图主要的交互方法包括交互提示信息和数据过滤选择。

其中，交互提示信息可在鼠标悬停时展示具体的数据数值，同时会将当前选定的国家高亮显示，如图 3.11 所示。

数据过滤选择主要依赖于左侧的数据范围条，用户可以将鼠标移到数据范围条上并指定要查看的条件，例如指定条件为出现 1 次，此时地图会将举办次数为 1 的国家高亮显示，其他地区保持不变，如图 3.12 所示。

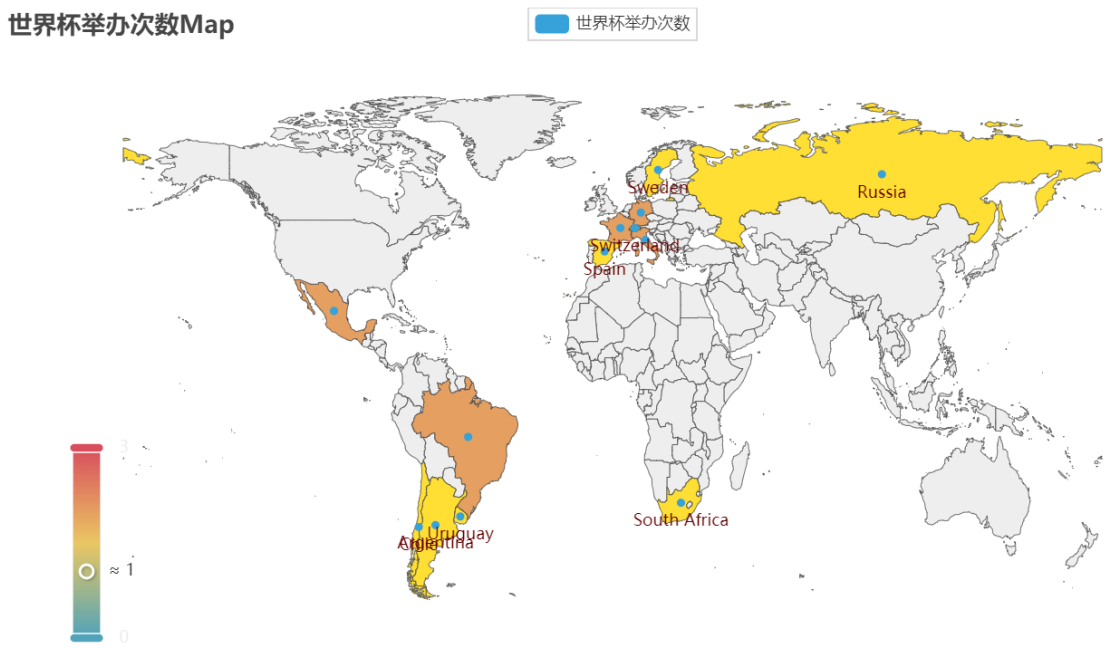

图 3.12：地图视图——世界杯举办国家分布地图用户交互效果

（3）视图分析

本地图展示了世界各国举办世界杯的次数分布情况。通过观察地图，可以明显看出各国在世界杯举办次数上的差异，有些国家举办了多次世界杯，而有些国家却从未有过。例如巴西、墨西哥、法国、意大利、德国均举办过 2 次世界杯，俄罗斯、阿根廷等地举办过 1 次世界杯，而其余绝大多数国家都从未举办过世界杯。因此世界杯的举办地区分布是极度不平衡的。

根据本地图，用户可以直观地了解到各国家的举办次数分布情况，为深入了解世界杯的历史提供了重要参考。

3.8 散点图视图：主客场进球数量出现次数散点图（气泡图）

（1）设计思路及设计过程

散点图用于展示离散数据的分布情况。在这里，我使用散点图（气泡图）展示世界杯比赛中主客场进球数量出现的次数分布情况，通过设置散点的位置、大小以及散点的颜色来表达数据。图中，横轴和纵轴分别代表主队和客队的进球数量，而散点的位置则代表了这种进球数量组合的出现次数，散点的大小和颜色代表了这种组合的频次，散点越大、颜色越红代表出现次数越多。据此，用户可以直观地看到世界杯比赛中各种进球数量组合的次数情况。

可视化编码：本视图采用了散点作为图形元素，使用了位置、大小和颜色等

视觉通道来表达数据。根据主客场进球组合的出现次数，使用了 Rainbow 颜色映射，通过设置散点的大小和颜色来表示数据的频次。

代码实现：首先从数据集中选取了'Home Team Goals'和'Away Team Goals'两列进行处理，对数据进行分组统计，计算每种主客场进球组合出现的次数。接下来，利用 Matplotlib 库绘制散点图，其中散点的 x 轴和 y 轴分别代表主队进球数和客队进球数，散点的大小和颜色由出现次数决定。颜色主要使用了 rainbow 调色板，并通过颜色深浅展示不同次数的组合。此外，还设置了颜色范围的最小值 cmin 和最大值 cmax，将颜色的映射范围限定在[0, 10]之间。最后，设置图表标签和标题。

本视图如图 3.13 所示。

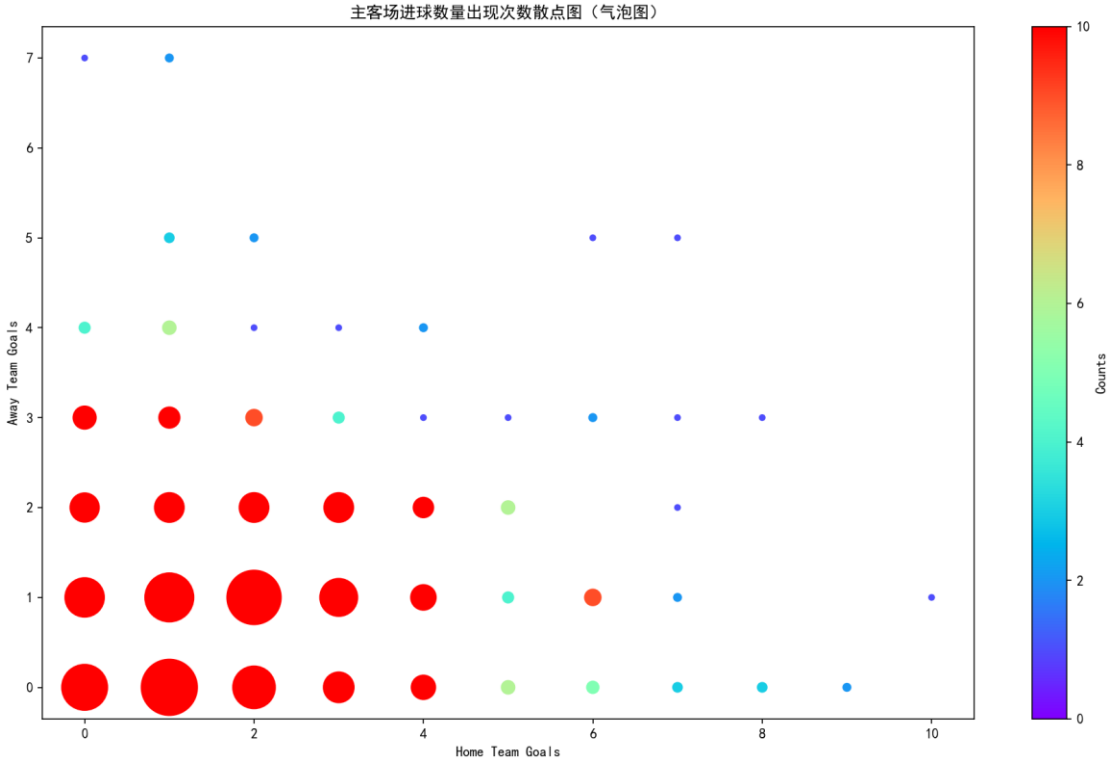

图 3.13：散点图视图——主客场进球数量出现次数散点图（气泡图）

（2）交互设计

本视图没有典型的交互设计，因为 Matplotlib 库绘制的图是静态的。

（3）视图分析

该散点图展示了世界杯比赛中主客场进球数量出现的次数分布情况，用户可以通过散点图了解各种进球数量组合的出现频次。从图中可以观察到不同数量的进球组合在比赛中的频次情况，其中左下角的点都很大，而左侧上方和右侧下方的点都很小甚至不存在。这说明比分相差不大的情况出现的很多，而比分相差悬

殊的情况极为少见。同时还可以发现，整个图关于 y=x 这条直线基本对称，说明世界杯比赛中主客场的影响不大，较为公平。

综上所述，用户不仅可以从横纵坐标获得信息，还可以通过颜色深浅和点的大小获得更直观的数据表达。

3.9 词云图视图：世界杯球员名称词云图

（1）设计思路及设计过程

词云图可以用于直观地展示出现频率高的词语，因此在世界杯数据集中非常合适。为了列举出世界杯历史上“最有影响力的球员”，我采用了词云图的方式来展示 WorldCupPlayers.csv 数据集中球员名称的出现频率。在词云图中，球员名称出现次数较多的会显得更加突出，可以反映球员的重要程度。

可视化编码：图形元素为一个词云图对象；采用的视觉通道包括不同的文字大小、不同的排布方向、丰富的色调（红蓝绿紫棕）。

代码实现：首先使用 value_counts()方法统计各个球员的出场次数，接着利用 WordCloud 模块创建一个词云图，将球员名和出场次数作为数据对添加到词云中。然后设置全局选项，包括词云图的标题和提示框选项。

本视图如图 3.14 所示。

图 3.14：词云图视图——世界杯球员名称词云图

（2）交互设计

本视图主要的交互方法是交互提示信息。在鼠标移到某个球员名称上时，会

展示某个球员具体的出现次数。

（3）视图分析

词云图中文本的大小反映了该词的相对重要程度，因此可以得出哪些球员名称在数据集中出现的频次较高。尺寸越大的球员名称代表着其在数据集中出现的次数越多，说明其重要性或出现频率较高。

从图中可以看出，世界杯历史上影响力最大的球员包括RONALDO、OSCAR、MULLER、LEAO、MESSI 等人。这可以帮助用户找出历史上最优秀的球员。

3.10 水球图视图：前九名队伍在世界杯历史进球总数的占比水球图

（1）设计思路及设计过程

水球图适合展示数据之间的比例关系，通过图中的水位可以判断所占的相对比例。为了找出进球最多的前九支队伍的进球数在世界杯历史进球总数中的比例，我采用了水球图的视图进行展示。本图中，排名第一的队伍的水位设为 100%，排名第二至第九的队伍的水位高度代表其与第一名的进球数之比。

可视化编码：本视图的图形元素为二维“面”和一维“线”，分别是水球图呈现出的圆形和水位分隔处的曲线；采用的视觉通道包括圆形的形状、蓝色的色调、不同的亮度（深色和浅色）。

代码实现：首先，针对某一个特定的队伍，使用 groupby 和 sum 函数对其在主队和客队的进球数求和，得到每支队伍的总进球数。随后将主队和客队总进球数这两个 DataFrame 进行拼接，得到每支队伍的总进球数。然后再次使用 groupby 和 sum 函数得到了全世界总的进球数。接着计算前九名进球总数最多的队伍，并计算它们的进球总数占比。之后，使用 pyecharts 中的 Liquid()函数来创建多个水球图，使用循环对前九名队伍的进球总数占比进行可视化，同时通过 set_global_opts 设置了图表标题等选项。最后仍然使用第 3.5 节中的 page 机制将这些水球图集中到一个页面中。

本视图如图 3.15 所示。

图 3.15：水球图视图——前九名队伍在世界杯历史进球总数的占比水球图

（2）交互设计

本视图主要的交互方法是交互提示信息。在鼠标移到某个水球图上时，会展示某个队伍在世界杯进球总数中的占比情况，同时会显示具体的进球总数量，便于获取隐藏在比例之下的信息。

（3）视图分析

本视图包含了多个水球图，分别展示了世界杯进球最多的前九支队伍的进球数在世界杯历史上进球总数中的占比。通过每个水球图，用户可以直观地看出每支队伍在整个世界杯历史上的进球数占比情况，从而了解历届世界杯中表现最佳的前九支队伍。同时，由于进球数最高的队伍被作为 100%的比例标准，因此用户也可以参照排名第一的队伍，观察排名第二至第九的队伍与排名第一的队伍的差距。

4 设计总结与心得

4.1 实验总结

在本次数据可视化实验中，我针对世界杯数据集完成了数据分析、数据预处理、可视化视图选择与实现、用户交互设计四个阶段，提取出其中的有效信息，基本完整地表现了整个数据集的内容。整体上，我总共设计了 9 个不同的可视化视图，针对特定的场景使用 matplotlib 和 pyecharts 库完成了图像的绘制。

具体来说，我设计的可视化视图分别是：

- **分组柱状图视图**：世界杯进球数和参赛队伍数柱状图
- **折线图视图**：观众人数和比赛场次随时间变化趋势折线图
- **面积图视图**：观众人数和比赛场次随时间变化趋势面积图
- **环形图视图**：世界杯结果分布环形图
- **多层级饼图（南丁格尔玫瑰图）视图**：世界杯结果分布饼图
- **地图视图**：世界杯举办国家分布地图
- **散点图视图**：主客场进球数量出现次数散点图（气泡图）
- **词云图视图**：世界杯球员名称词云图
- **水球图视图**：前九名队伍在世界杯历史进球总数的占比水球图

4.1.1 遇到的问题及处理

本次实验还是遇到了一些问题，主要是两方面造成的：一方面，之前两个实验都是给定任务下的实验，不需要进行过多的方案设计与思考，发挥空间很小，而本实验则完全需要自己思考数据可视化方案，反复权衡才可能达到较好的效果；另一方面，在学习本课程之前，我从来没有接触过 Pyecharts 库，因此遇到了不少语法错误，后来通过反复查阅资料、请教老师和同学终于得以解决。

整个实验我遇到的典型问题有下面几项：

- 在“观众人数和比赛场次随时间变化趋势折线图”中，一开始没有关注两个属性数据的数量级差异，只使用了一个轴，导致观众人数的显示正常，而比赛场次的折线非常贴近于 x 轴（因为数据相比观众人数太少了，看起来像一条接近于 0 的直线）。发现这一问题后，我通过调用 extend_axis()方法添加了一个额外的纵轴，解决了显示异常的问题。

- 在“世界杯结果分布环形图”中，最初使用了 Grid 调节四个子图的位置，主要是通过修改 pos_left、pos_right、pos_top 和 height 四个参数。然而在合理的

参数下（四组分别为 0.1/0.1/0.5/0.45、0.1/0.6/0.1/0.45、0.55/0.1/0.5/0.45、0.55/0.6/0.1/0.45），四个子图仍然存在一定的重叠现象。后来我查阅了 Pyechart 库中的 page 库函数用法，发现实现更简单并且效果很稳定，于是将四个子图放在一个 page 后完美解决了这一问题，观感很好。

• 在“世界杯举办国家分布地图”中，一开始没有统计各国举办次数的最大值，导致颜色变化极不均匀（颜色从深蓝色到红色的映射范围是[0, 100]，但举办次数的最大值为 2，导致所有国家要么是 0 次显示为白色，要么显示为深蓝色，完全无法看出任何差异）。后来经过统计，将映射范围改为[0, 2]，就解决了这一问题。在后面的“主客场进球数量出现次数散点图（气泡图）”中，我提前完成了一轮统计，从而避免踩坑。

4.1.2 设计方案存在的不足

• 虽然我设计了 9 种不同的视图，但还是没有能彻底描述整个数据集中的信息。还可以深入挖掘的信息有很多，例如教练的执教水平（可以通过分析其执导过的球员的进球数、事件数、入围决赛的次数等指标完成）、秋衣号码与球员影响力之间的相关性等等。但是这些信息涉及到的因素过多，且需要引入额外的信息进行更全面的分析，因此本实验中我没有深入探讨这些问题。

• 可视化程序未能实现实时联网爬取数据。由于世界杯的数据随时间变化会不断更新，因此如果在若干年后想要再次使用本可视化方案进行数据可视化，则可能需要重新获取数据。

• 各视图之间的连贯性较弱，可能还需要引入一种整体的数据讲述模式，避免各个视图过于独立的问题。这样可以更好地向观众传达一种数据故事，将各个视图有机地结合在一起，呈现出清晰、连贯的数据故事线索。这也是数据可视化的初衷。

4.2 实验心得

在对世界杯数据集进行数据可视化的过程中，我深刻体会到了数据可视化在揭示数据特征和提供洞察力方面的重要性。通过对整体设计框架和各个阶段的实际操作，我总结了以下几点心得体会：

（1）数据分析与预处理的重要性

数据分析和预处理是数据可视化过程中至关重要的一环，如果没有好的数据，可视化的结果就是低质量的。

首先，要做好数据分析，这可以帮助确定可选的合适视图，例如世界杯数据集完全不适合使用桑基图（Sankey），因为它不存在任何的“能量流动现象”。

在进行数据预处理时，需要仔细观察数据特征，检测并处理可能存在的噪声

数据，以确保后续可视化呈现的准确性和可信度。以世界杯数据集为例，如果某一列数据存在缺失的问题，那么统计时必定会出现错误；如果年份（Year）字段不是整数而是浮点数，就无法被正确地读取，更无法被正确地录入坐标轴。可见，数据预处理也是相当重要的。

（2）可视化编码与视图选择

在可视化编码阶段，需要深入分析数据特征，选择合适的可视化编码（标记和视觉通道）和视图类型来展现数据。

对于可视化编码，实际上对任何一种情形都会面临很丰富的选择，例如标记（图形元素）有点、线、面等选择，通道有位置、大小、形状、方向、色调、饱和度、亮度等大量角度需要考虑。因此需要针对我们实际处理的数据特征选取最优的编码。

同时，我们还需要思考自己到底想表现出数据的什么信息，并以直观易懂的方式呈现数据。例如：

- 对于离散型数据，可以使用散点图，采用点、位置等可视化编码
- 对于具有明显时间变迁特点的序列数据，可以使用折线图，采用点、线、连接关系等可视化编码
- 对于构成性数据，可以使用饼图或水球图，采用面、大小、色调等可视化编码

（3）用户交互设计的增强作用

引入用户交互设计（如滚动、缩放、选择、过滤等）能增强可视化图形的丰富度，提高用户对数据的互动体验和自由查询的便利性。这对于满足不同用户对数据分析需求的灵活性和个性化提供了有力支持。例如对于“世界杯进球数和参赛队伍数柱状图”，如果用户只希望查询进球数的相关数据，那么可以只保留进球数这一属性，忽略掉参赛队伍数的属性。

总体来说，本次实验的体验还是相当好的，因为此前我只会用 Matplotlib 库绘制一些图，这些图全部都是静态的，完全无法体现与使用者的交互，并且绘制的图像都是很常见的简单类型图。通过本次实验，我惊喜地发现数据可视化领域存在大量我从未接触过的视图和视觉通道，它们可以组合出无限的可能，更加深刻地刻画数据的特征，这让我在设计数据可视化方案、使用数据可视化技术两个维度上有了大幅度的提高，也丰富了我的实践经验。

4.3 意见与建议

本次实验我认为设计的很全面，在实验内容上我没有任何意见。当然，从结果导向的角度来说，如果还有优秀作品交流讨论环节就更好了，因为每个人是从四个数据集中选择一个完成设计方案，对于其它数据集的内核了解较少，如果能展示一下各类数据集的优秀作品，或许还能更充分地拓宽大家的视野。（不过不知道可操作性大不大）

原创性声明

本人郑重声明本报告内容，是由作者本人独立完成的。有关观点、方法、数据和文献等的引用已在文中指出。除文中已注明引用的内容外，本报告不包含任何其他个人或集体已经公开发表的作品成果，不存在剽窃、抄袭行为。

已阅读并同意以下内容。

判定为不合格的一些情形：

（1） 请人代做或冒名顶替者；

（2） 替人做且不听劝告者；

（3） 实验报告内容抄袭或雷同者；

（4） 实验报告内容与实际实验内容不一致者；

（5） 实验代码抄袭者。

作者签名：

附录一 实验代码

• 分组柱状图视图：世界杯进球数和参赛队伍数柱状图

```
from pyecharts.charts import Bar
import pandas as pd
from pyecharts import options as opts

world_cup_data = pd.read_csv('WorldCupsSummary.csv')

bar = (
    Bar()
    .add_xaxis(world_cup_data['Year'].tolist())
    .add_yaxis("进球数", world_cup_data['GoalsScored'].tolist())
    .add_yaxis("参赛队伍数", world_cup_data['QualifiedTeams'].tolist())
    .set_global_opts(title_opts=opts.TitleOpts(title="不同年份进球数与参赛队伍数的柱状图"),
                     datazoom_opts=[opts.DataZoomOpts(type_="slider", range_start=0, range_end=100)])
)

bar.render('世界杯进球数和参赛队伍数柱状图.html')
```

• 折线图视图：观众人数和比赛场次随时间变化趋势折线图

```
from pyecharts.charts import Line
import pandas as pd
from pyecharts import options as opts

world_cup_data = pd.read_csv('WorldCupsSummary.csv')

line = (
    Line()
    .add_xaxis(world_cup_data['Year'].astype("str").tolist())
    .add_yaxis("观众人数", world_cup_data['Attendance'].tolist())
    .extend_axis(
        yaxis=opts.AxisOpts(
            name="比赛场次",
            type_="value"
        )
    )
    .add_yaxis("比赛场次", world_cup_data['MatchesPlayed'].tolist(), yaxis_index=1)
    .set_global_opts(
        title_opts=opts.TitleOpts(title="观众人数和比赛场次随时间变化趋势折线图"),
        yaxis_opts=opts.AxisOpts(
            type_="value",
```

```
                name="观众人数"
            ),
            tooltip_opts=opts.TooltipOpts(trigger="axis", axis_pointer_type="cross"),
            datazoom_opts=[opts.DataZoomOpts(type_="slider", range_start=0,
range_end=100)]
        )
)

line.render('观众人数和比赛场次随时间变化趋势折线图.html')
```

- 面积图视图：观众人数和比赛场次随时间变化趋势面积图

```
from pyecharts.charts import Line
import pandas as pd
from pyecharts import options as opts

world_cup_data = pd.read_csv('WorldCupsSummary.csv')

line = (
    Line()
    .add_xaxis(world_cup_data['Year'].astype("str").tolist())
    .add_yaxis("观众人数", world_cup_data['Attendance'].tolist(),
areastyle_opts=opts.AreaStyleOpts(opacity=0.5))
    .extend_axis(
        yaxis=opts.AxisOpts(
            name="比赛场次",
            type_="value"
        )
    )
    .add_yaxis("比赛场次", world_cup_data['MatchesPlayed'].tolist(), yaxis_index=1,
areastyle_opts=opts.AreaStyleOpts(opacity=0.5))
    .set_global_opts(
        title_opts=opts.TitleOpts(title="观众人数和比赛场次随时间变化趋势面积图"),
        yaxis_opts=opts.AxisOpts(
            type_="value",
            name="观众人数"
        ),
        tooltip_opts=opts.TooltipOpts(trigger="axis", axis_pointer_type="cross"),
        datazoom_opts=[opts.DataZoomOpts(type_="slider", range_start=0,
range_end=100)]
    )
)

line.render('观众人数和比赛场次随时间变化趋势面积图.html')
```

- 环形图视图：世界杯结果分布环形图

```
from pyecharts import options as opts
from pyecharts.charts import    Page, Pie
import pandas as pd

world_cup_data = pd.read_csv('WorldCupsSummary.csv')

champions = world_cup_data['Winner'].value_counts()
runners_up = world_cup_data['Second'].value_counts()
third_places = world_cup_data['Third'].value_counts()
fourth_places = world_cup_data['Fourth'].value_counts()

pie1 = (
    Pie()
    .add(
        "冠军",
        [list(z) for z in zip(champions.index.tolist(), champions.values.tolist())],
        radius=["50%", "70%"]   # 设置圆环图的内外半径
    )
    .set_series_opts(label_opts=opts.LabelOpts(formatter="{b}: {c}"))
    .set_global_opts(
        title_opts=opts.TitleOpts(title="冠军分布"),
        legend_opts=opts.LegendOpts(pos_left="right", orient="vertical")
    )
)

pie2 = (
    Pie()
    .add(
        "亚军",
        [list(z) for z in zip(runners_up.index.tolist(), runners_up.values.tolist())],
        radius=["50%", "70%"]
    )
    .set_series_opts(label_opts=opts.LabelOpts(formatter="{b}: {c}"))
    .set_global_opts(
        title_opts=opts.TitleOpts(title="亚军分布"),
        legend_opts=opts.LegendOpts(pos_left="right", orient="vertical")
    )
)

pie3 = (
    Pie()
    .add(
        "季军",
```

```
            [list(z) for z in zip(third_places.index.tolist(), third_places.values.tolist())],
            radius=["50%", "70%"]
        )
        .set_series_opts(label_opts=opts.LabelOpts(formatter="{b}: {c}"))
        .set_global_opts(
            title_opts=opts.TitleOpts(title="季军分布"),
            legend_opts=opts.LegendOpts(pos_left="right", orient="vertical")
        )
)

pie4 = (
        Pie()
        .add(
            "第 4 名",
            [list(z) for z in zip(fourth_places.index.tolist(), fourth_places.values.tolist())],
            radius=["50%", "70%"]
        )
        .set_series_opts(label_opts=opts.LabelOpts(formatter="{b}: {c}"))
        .set_global_opts(
            title_opts=opts.TitleOpts(title="第 4 名分布"),
            legend_opts=opts.LegendOpts(pos_left="right", orient="vertical")
        )
)

page = Page(page_title='世界杯结果分布', layout=Page.SimplePageLayout)
page.add(pie1, pie2, pie3, pie4)

page.render('世界杯结果分布-环形图.html')
```

- 多层级饼图（南丁格尔玫瑰图）视图：世界杯结果分布饼图

```
from pyecharts import options as opts
from pyecharts.charts import Grid, Pie
import pandas as pd

world_cup_data = pd.read_csv('WorldCupsSummary.csv')

champions = world_cup_data['Winner'].value_counts()
runners_up = world_cup_data['Second'].value_counts()
third_places = world_cup_data['Third'].value_counts()
fourth_places = world_cup_data['Fourth'].value_counts()

data_champions = list(zip(champions.index.tolist(), champions.values.tolist()))
data_runners_up = list(zip(runners_up.index.tolist(), runners_up.values.tolist()))
data_third_places = list(zip(third_places.index.tolist(), third_places.values.tolist()))
```

```
data_fourth_places = list(zip(fourth_places.index.tolist(), fourth_places.values.tolist()))

pie1 = (
    Pie()
    .add(
        "冠军",
        data_champions,
        radius=["50%", "70%"],
        rosetype="radius"
    )
    .set_series_opts(label_opts=opts.LabelOpts(formatter="{b}: {c}"))
    # .set_global_opts(title_opts=opts.TitleOpts(title="世界杯冠军、亚军、季军、第 4 名国家分布图"))
)

pie2 = (
    Pie()
    .add(
        "亚军",
        data_runners_up,
        radius=["40%", "60%"],
        rosetype="radius"
    )
    .set_series_opts(label_opts=opts.LabelOpts(formatter="{b}: {c}"))
    .set_global_opts(legend_opts=opts.LegendOpts(is_show=False))
)

pie3 = (
    Pie()
    .add(
        "季军",
        data_third_places,
        radius=["30%", "50%"],
        rosetype="radius"
    )
    .set_series_opts(label_opts=opts.LabelOpts(formatter="{b}: {c}"))
    .set_global_opts(legend_opts=opts.LegendOpts(is_show=False))
)

pie4 = (
    Pie()
    .add(
        "第 4 名",
        data_fourth_places,
```

```
            radius=["20%", "40%"],
            rosetype="radius"
        )
        .set_series_opts(label_opts=opts.LabelOpts(formatter="{b}: {c}"))
        .set_global_opts(legend_opts=opts.LegendOpts(is_show=False))
)

grid = (
        Grid()
        .add(pie1, grid_opts=opts.GridOpts(pos_left="10%", pos_right="35%"),
is_control_axis_index=True)
        .add(pie2, grid_opts=opts.GridOpts(pos_left="30%", pos_right="15%"),
is_control_axis_index=True)
        .add(pie3, grid_opts=opts.GridOpts(pos_left="50%", pos_right="15%"),
is_control_axis_index=True)
        .add(pie4, grid_opts=opts.GridOpts(pos_left="70%", pos_right="0%"),
is_control_axis_index=True)
)
grid.render("世界杯结果分布-多层级饼图.html")
```

- **地图视图：世界杯举办国家分布地图**

```
from pyecharts.globals import ThemeType
import pandas as pd
from pyecharts.charts import Map
from pyecharts import options as opts

world_cup_data = pd.read_csv('WorldCupsSummary.csv')

host_count = world_cup_data['HostCountry'].value_counts().reset_index()
host_count.columns = ['HostCountry', 'Count']

map_ = (
        Map(init_opts=opts.InitOpts(theme=ThemeType.LIGHT))
        .add("世界杯举办次数", [list(z) for z in zip(host_count['HostCountry'],
host_count['Count'])], "world")
        .set_global_opts(title_opts=opts.TitleOpts(title="世界杯举办次数 Map"),
                         visualmap_opts=opts.VisualMapOpts(is_show=True,
                                                           min_=0, max_=3,
                                                           pieces=[{"min": 0,
"max": 0, "label": "0 次"},
                                                                   {"min": 1,
"max": 1, "label": "1 次"},
                                                                   {"min": 2,
"max": 2, "label": "2 次"},
```

```
                                                    {"min": 3,
"max": 3, "label": "3 次"},
                                                    ],
                                                pos_left="5%",
pos_bottom="5%"))
        .set_series_opts(label_opts=opts.LabelOpts(is_show=False))
)

map_.render('世界杯举办国家分布图-地图.html')
```

- **散点图视图：主客场进球数量出现次数散点图（气泡图）**

```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('WorldCupMatches.csv')

df_clean = df[['Home Team Goals', 'Away Team Goals']].dropna()
df_clean['Home Team Goals'] = df_clean['Home Team Goals'].astype(float)
df_clean['Away Team Goals'] = df_clean['Away Team Goals'].astype(float)

counts = df_clean.groupby(['Home Team Goals', 'Away Team
Goals']).size().reset_index(name='Counts')

x = counts['Home Team Goals']
y = counts['Away Team Goals']
sizes = counts['Counts']

# 设置散点图颜色范围
cmin, cmax = 0, 10

# 设置颜色数组
colors = counts['Counts']

plt.scatter(x, y, s=15*sizes, c=colors, cmap='rainbow', vmin=cmin, vmax=cmax)
plt.colorbar(label='Counts')
plt.xlabel('Home Team Goals')
plt.ylabel('Away Team Goals')
plt.title('主客场进球数量出现次数散点图（气泡图）')

plt.rcParams['font.sans-serif'] = ['SimHei']
plt.show()
```

- **词云图视图：世界杯球员名称词云图**

```
import pandas as pd
from pyecharts import options as opts
from pyecharts.charts import WordCloud

data = pd.read_csv('WorldCupPlayers.csv')

player_counts = data['Player Name'].value_counts()

wordcloud = (
    WordCloud()
    .add(series_name="", data_pair=player_counts.items())
    .set_global_opts(
        title_opts=opts.TitleOpts(title="Player Name 词云图"),
        tooltip_opts=opts.TooltipOpts(trigger="item"),
    )
)

wordcloud.render("Player Name 词云图.html")
```

- 水球图视图：前九名队伍在世界杯历史进球总数的占比水球图

```
import pandas as pd
import numpy as np
from pyecharts import options as opts
from pyecharts.charts import Grid, Liquid, Page

df = pd.read_csv('WorldCupMatches.csv')

team_goals = pd.concat([df.groupby('Home Team Name')['Home Team Goals'].sum(),
df.groupby('Away Team Name')['Away Team Goals'].sum()])
total_goals = team_goals.groupby(team_goals.index).sum()

top_9_teams = total_goals.sort_values(ascending=False)[:9]
total = total_goals.sum()
percentages = top_9_teams / total

liquids = []
for team, percentage in zip(top_9_teams.index, percentages):
    liquid = (
        Liquid()
        .add(
            "",
            [np.round(float(percentage) * 100, 2)],
            label_opts=opts.LabelOpts(formatter="{c}%"),
        )
```

```
            .set_global_opts(
                title_opts=opts.TitleOpts(
                    title="{} 进球数在世界杯进球总数的占比".format(team)
                )
            )
        )
        liquids.append(liquid)

page = Page(page_title="进球数在世界杯进球总数的占比",
layout=Page.SimplePageLayout)
for liquid in liquids:
        page.add(liquid)

page.render("liquid_chart.html")
```